Package 'TSSr'

December 15, 2025

Type Package

Version 0.99.8

Date 2021-09-26

Title TSS sequencing data analysis

Description TSSr package provides a comprehensive workflow on TSS data starts from identification of accurate TSS locations, clustering TSSs within small genomic regions corresponding to core promoters, and transcriptional activity quantifications, as well as specialized downstream analyses including core promoter shape, cluster annotation, gene differential expression, core promoter shift. TSSr can take multiple formats of files as input, such as Binary Sequence Alignment Map (BAM) files (single-ended or paired-ended), Browser Extension Data (bed) files, BigWig files, ctss files or tss tables. TSSr also generates various types of TSS or core promoter track files which can be visualized in the UCSC Genome Browser or Integrative Genomics Viewer (IGV). TSSr also exports downstream analyses result tables and plots. Multiple cores are supported on Linux or Mac platforms.

License MIT + file LICENSE

Encoding UTF-8

LazyData FALSE

Depends R (>= 4.1.0)

Imports stringr(>= 1.4.0), rtracklayer(>= 1.50.0),ggplot2(>= 3.3.5), GenomicFeatures(>= 1.42.3), Gviz(>= 1.34.1), DESeq2(>= 1.30.1), calibrate(>= 1.7.7), ggfortify(>= 0.4.12), IRanges(>= 2.24.1), GenomicRanges(>= 1.42.0), Rsamtools(>= 2.6.0), GenomicAlignments(>= 1.26.0), GenomeInfoDb(>= 1.26.7),dplyr(>= 1.0.7), BiocGenerics(>= 0.36.1), data.table(>= 1.14.0), methods(>= 4.0.3), grDevices(>= 4.0.3), graphics(>= 4.0.3), stats(>= 4.0.3), utils(>= 4.0.3),parallel(>= 4.0.3), rmarkdown

Suggests knitr, pkgdown, testthat (>= 3.0.0), BSgenome.Scerevisiae.UCSC.sacCer3

Config/testthat/edition 3

biocViews Software, Transcription, Coverage, GeneExpression, GeneRegulation, PeakDetection, DataImport, DataRepresentation, Transcriptomics, Sequencing, Annotation, GenomeBrowsers, Normalization, Preprocessing, Visualization,DifferentialExpression, Alignment,Clustering

URL https://github.com/Linlab-slu/TSSr

BugReports https://github.com/Linlab-slu/TSSr/issues

VignetteBuilder knitr

RoxygenNote 7.1.2

NeedsCompilation no

Author Zhaolian Lu [aut, com] (<https://orcid.org/0000-0001-5002-7007>),
Keenan Berry [aut, com],
Zhenbin Hu [aut, ctb],
Yu Zhan [aut, ctb],
Tae-Hyuk (Ted) Ahn [aut, cph],
Zhenguo Lin [aut, cre, cph] (<https://orcid.org/0000-0002-8400-9138>),
National Science Foundation [fnd] (NSF 1951332)

Maintainer Zhenguo Lin <zhenguo.lin@slu.edu>

R topics documented:

TSSr-package *TSSr: A package for transcription start site sequencing data analyses.*

Description

TSSr is designed to analyze transcription start sites (TSSs) and core promoters with most types of 5'end sequencing data, such as cap analysis of gene expression (CAGE) (Takahashi, Lassmann et al. 2012), no-amplification non-tagging CAGE libraries for Illumina next-generation sequencers (nAnT-iCAGE) (Murata, Nishiyori-Sueki et al. 2014), a Super-Low Input Carrier-CAGE (SLIC-CAGE) (Cvetesic, Leitch et al. 2018), NanoCAGE (Cumbie, Ivanchenko et al. 2015), TSS-seq (Malabat, Feuerbach et al. 2015), transcript isoform sequencing (TIF-seq) (Pelechano, Wei et al. 2013), transcript-leaders sequencing (TL-seq) (Arribere and Gilbert 2013), precision nuclear run-on sequencing (PRO-Cap) (Mahat, Kwak et al. 2016), and GRO-Cap/5'GRO-seq (Kruesi, Core et al. 2013).

Details

TSSr package provides a comprehensive workflow on TSS data starts from identification of accurate TSS locations, clustering TSSs within small genomic regions corresponding to core promoters, and transcriptional activity quantifications, as well as specialized downstream analyses including core promoter shape, cluster annotation, gene differential expression, core promoter shift. TSSr can take multiple formats of files as input, such as Binary Sequence Alignment Map (BAM) files (single-ended or paired-ended), Browser Extension Data (bed) files, BigWig files, ctss files or tss tables. TSSr also generates various types of TSS or core promoter track files which can be visualized in the UCSC Genome Browser or Integrative Genomics Viewer (IGV). TSSr also exports downstream analyses result tables and plots. Multiple cores are supported on Linux or Mac platforms.

---

annotateCluster *Annotate clusters with GFF annotation file.*

Description

Annotates clusters with gene or transcript names from GFF annotation file.

Usage

```
annotateCluster(object,clusters = "consensusClusters",filterCluster = TRUE
, filterClusterThreshold = 0.02, annotationType = "genes",upstream=1000
, upstreamOverlap = 500,downstream = 0)

## S4 method for signature 'TSSr'
annotateCluster(
  object,
  clusters = "consensusClusters",
  filterCluster = TRUE,
  filterClusterThreshold = 0.02,
  annotationType = "genes",
  upstream = 1000,
  upstreamOverlap = 500,
  downstream = 0
)
```

Arguments

object	A TSSr object
clusters	Clusters to be annotated: "consensusClusters" or "tagClusters". Default is "consensusClusters".
filterCluster	Logical indicating whether clusters downstream of a highly expressed cluster are filtered. Setting filterCluster as "TRUE" would reduce weak clusters brought from recapping, transcriptional or sequencing noise. Default is TRUE.
filterClusterThreshold	Ignore downstream clusters if signal < filterClusterThreshold*the strongest clusters within the same gene promoter region. Default value = 0.02.
annotationType	Specify annotation feature to be associated with: "genes" or "transcripts". Default is "genes".
upstream	Upstream distance to the start position of annotation feature. Default value = 1000.
upstreamOverlap	Upstream distance to the start position of annotation feature if overlapped with the upstream neighboring feature. Default value = 500.
downstream	Downstream distance to the start position of annotation feature. Default value = 0. Note: if annotationType == "transctipt" or the gene annotations start from transcription start sites (TSSs), the recommended value = 500.

Value

Large List of elements - one element for each sample

Examples

```
data(exampleTSSr)
annotateCluster(exampleTSSr,clusters = "consensusClusters", filterCluster = TRUE
, filterClusterThreshold = 0.02, annotationType = "genes", upstream=1000
, upstreamOverlap = 500, downstream = 0)
```

---

callEnhancer *Identification of enhancers*

Description

Calculates enhancer candidates, which are characterized by bidirectional clusters as described in Andersson et al. 2014.

Usage

```
callEnhancer(object, flanking = 400, dis2gene = 2000)

## S4 method for signature 'TSSr'
callEnhancer(object, flanking = 400, dis2gene = 2000)
```

Arguments

object	A TSSr object.
flanking	The flanking region range where bidirectional clusters composing a enhancer candidate. Default is 400.
dis2gene	The minimum distance to the main annotated core promoter of genes. Default is 2000.

Value

Large List of elements - one element for each sample

Examples

```
data(exampleTSSr)
#callEnhancer(exampleTSSr,flanking = 400,dis2gene=2000)
```

---

clusterTSS *Cluster TSSs into tag clusters*

Description

Clusters TSSs within small genomic regions into tag clusters (TCs) using "peakclu" method. "peakclu" method is an implementation of peak-based clustering. The minimum distance of two neighboring peaks can be specified.

Usage

```
clusterTSS(object, method = "peakclu", peakDistance=100,extensionDistance=30
, localThreshold = 0.02,clusterThreshold = 1, useMultiCore=FALSE, numCores=NULL)

## S4 method for signature 'TSSr'
clusterTSS(
  object,
  method = "peakclu",
  peakDistance = 100,
  extensionDistance = 30,
  localThreshold = 0.02,
  clusterThreshold = 1,
  useMultiCore = FALSE,
  numCores = NULL
)
```

Arguments

object	A TSSr object
method	Clustering method to be used for clustering: "peakclu". Default is "peakclu".
peakDistance	Minimum distance of two neighboring peaks. Default value = 100.

`extensionDistance`
: Maximal distance between peak and its neighboring TSS or two neighboring TSSs to be grouped in the same cluster. Default value = 30.

`localThreshold`
: Ignore downstream TSSs with signal < localThreshold*peak within clusters, which is used to filter TSS signals brought from possible recapping events, or sequencing noise. Default value = 0.02.

`clusterThreshold`
: Ignore clusters if signal < clusterThreshold. Default value = 1.

`useMultiCore`
: Logical indicating whether multiple cores are used (TRUE) or not (FALSE). Default is FALSE.

`numCores`
: Number of cores are used in clustering step. Used only if useMultiCore = TRUE. Default is NULL.

Value

Large List of elements - one element for each sample

Examples

```
data(exampleTSSr)
clusterTSS(exampleTSSr, method = "peakclu",clusterThreshold = 1,
useMultiCore=FALSE, numCores = NULL)
```

---

consensusCluster *Make consensus clusters across multiple samples.*

Description

Makes consensus clusters from multiple samples in TSSr object and calculates inter-quantile positions within consensus clusters for each sample.

Usage

```
consensusCluster(object, dis = 50
, useMultiCore=FALSE, numCores = NULL)

## S4 method for signature 'TSSr'
consensusCluster(object, dis = 50, useMultiCore = FALSE, numCores = NULL)
```

Arguments

`object`
: A TSSr object.

`dis`
: Minimum distance between two peaks to be aggregated together into the same consensus cluster.

`useMultiCore`
: Logical indicating whether multiple cores are used (TRUE) or not (FALSE). Default is FALSE.

`numCores`
: Number of cores are used in clustering step. Used only if useMultiCore = TRUE. Default is NULL.

Value

Large List of elements - one element for each sample

Examples

```
data(exampleTSSr)
consensusCluster(exampleTSSr,useMultiCore=FALSE)
```

---

deGene *Analysis of gene differential expression.*

Description

Analyzes gene-level differential expression using DESeq2 method (Love et al., 2014).

Usage

```
deGene(object,comparePairs=list(c("control","treat")), pval = 0.01,
 useMultiCore=FALSE, numCores = NULL)

## S4 method for signature 'TSSr'
deGene(
  object,
  comparePairs = list(c("control", "treat")),
  pval = 0.01,
  useMultiCore = FALSE,
  numCores = NULL
)
```

Arguments

object	A TSSr object.
comparePairs	Specified list of sample pairs for comparison with DESeq2 method.
pval	Genes with adjusted p value >= pVal will be returned. Default value = 0.01.
useMultiCore	Logical indicating whether multiple cores are used (TRUE) or not (FALSE). Default is FALSE.
numCores	Number of cores are used in clustering step. Used only if useMultiCore = TRUE. Default is NULL.

Value

Large List of elements - one element for each sample

Examples

```
data(exampleTSSr)
deGene(exampleTSSr,comparePairs=list(c("control","treat")), pval = 0.01)
```

exampleTSSr *TSSr example data*

Description

Subset of the CAGE dataset from the paper Lu and Lin, Genome Research 2019 Jul;29(7):1198-1210.

Usage

```
data(exampleTSSr)
```

Format

An object of class TSSr

exampleTSSr RangedSummarizedExperiment:TSSr

exportClustersTable *Export cluster tables*

Description

Export cluster tables to text files.

Usage

```
exportClustersTable(object, data = "assigned")

## S4 method for signature 'TSSr'
exportClustersTable(object, data = "assigned")
```

Arguments

object	A TSSr object.
data	Specify which cluster data will be exported: "tagClusters", "consensusClusters", "assigned", "unassigned". Default is "assigned".

Value

cluster tables

Examples

```
data(exampleTSSr)
#exportClustersTable(exampleTSSr, data = "tagClusters")
#exportClustersTable(exampleTSSr, data = "consensusClusters")
#exportClustersTable(exampleTSSr, data = "assigned")
#exportClustersTable(exampleTSSr, data = "unassigned")
```

exportClustersToBed *Creating bed files of clusters*

Description

Creates bed files of clusters.

Usage

```
exportClustersToBed(object, data = "consensusClusters", assigned = TRUE)

## S4 method for signature 'TSSr'
exportClustersToBed(object, data = "consensusClusters", assigned = TRUE)
```

Arguments

object	A TSSr object.
data	Specify which data will be exported: "tagClusters" or "consensusClusters". Default is "consensusClusters".
assigned	Specify which consensus clusters will be exported. Used only if data = "consensusClusters. Default is TRUE.

Value

bed files of clusters

Examples

```
data(exampleTSSr)
#exportTSStoBedgraph(exampleTSSr, data = "tagClusters")
#exportTSStoBedgraph(exampleTSSr, data = "consensusClusters")
```

exportDETable *Export gene differential expression results table*

Description

Exports gene differential expression results table to text files.

Usage

```
exportDETable(object, data = "sig")

## S4 method for signature 'TSSr'
exportDETable(object, data = "sig")
```

Arguments

object	A TSSr object.
data	Specify which data will be exported: "all" or "sig".

Value

gene differential expression tables

Examples

```
data(exampleTSSr)
#exportDETable(exampleTSSr, data="sig")
```

`exportEnhancerTable` *Export enhancer tables*

Description

Exports enhancer tables to text files.

Usage

```
exportEnhancerTable(object)

## S4 method for signature 'TSSr'
exportEnhancerTable(object)
```

Arguments

`object`	A TSSr object.

Value

enhancer candidate tables

Examples

```
data(exampleTSSr)
#exportEnhancerTable(exampleTSSr)
```

`exportShapeTable` *Export core promoter shape score tables*

Description

Exports core promoter shape score tables to text files. Shape score is calculated with shapeCluster(object) method.

Usage

```
exportShapeTable(object)

## S4 method for signature 'TSSr'
exportShapeTable(object)
```

Arguments

object	A TSSr object.

Value

core promoter shape score tables

Examples

```
data(exampleTSSr)
#exportShapeTable(exampleTSSr)
```

---

exportShiftTable *Export core promoter shift table*

Description

Export core promoter shift tables to text files.

Usage

```
exportShiftTable(object)

## S4 method for signature 'TSSr'
exportShiftTable(object)
```

Arguments

object	A TSSr object.

Value

core promoter shift tables

Examples

```
data(exampleTSSr)
#exportShiftTable(exampleTSSr)
```

exportTSStable *Export TSS tables*

Description

Exports TSS tables to text file.

Usage

```
exportTSStable(object, data = "raw", merged = "TRUE")

## S4 method for signature 'TSSr'
exportTSStable(object, data = "raw", merged = "TRUE")
```

Arguments

object	A TSSr object.
data	Specify which data will be exported: "raw" or "processed". Default is "raw".
merged	Specify whether to export merged TSS table. Used only if data = "raw".

Value

TSS tables

Examples

```
data(exampleTSSr)
#exportTSStable(exampleTSSr)
#exportTSStable(exampleTSSr, data="raw")
```

exportTSStoBedgraph *Creating Bedgraph/BigWig tracks of TSSs*

Description

Creates bedGraph/BigWig files of TSSs that can be visualized in the UCSC Genome Browser and Integrative Genomics Viewer (IGV).

Usage

```
exportTSStoBedgraph(object, data = "processed", format = "bedGraph",oneFile = FALSE)

## S4 method for signature 'TSSr'
exportTSStoBedgraph(
  object,
  data = "processed",
  format = "bedGraph",
  oneFile = FALSE
)
```

Arguments

object	A TSSr object.
data	Specify which data will be exported: "raw" or "processed". Default is "processed".
format	The format of output files: "bedGraph" or "BigWig". Default is "bedGraph".
oneFile	Logical, specify whether to export individual TSS tracks into the one bedGraph file (TRUE) of in separate bedGraph files (FALSE).

Value

Bedgraph/BigWig tracks of TSSs

Examples

```
data(exampleTSSr)
#exportTSStoBedgraph(exampleTSSr, data = "processed", format = "bedGraph")
```

---

filterTSS *Filter raw TSS counts or normalized TSS*

Description

Filters transcriptional or sequencing noise.

Usage

```
filterTSS(object, method = "poisson", normalization = TRUE,
pVal =0.01, tpmLow = 0.1)

## S4 method for signature 'TSSr'
filterTSS(
  object,
  method = "poisson",
  normalization = TRUE,
  pVal = 0.01,
  tpmLow = 0.1
)
```

Arguments

object	A TSSr object.
method	Method to be used for TSS filtering: "poisson" or "TPM". "poisson" can be used only if the input TSS data in raw number of counts.
normalization	Define whether normalization data to TPM. Used only if method = “poisson”. Default is TRUE.
pVal	Used only if method = "poisson". Default value is 0.01.
tpmLow	Used only if method = "TPM". Default value is 0.1.

Value

Large List of elements - one element for each sample

Examples

```
data(exampleTSSr)
filterTSS(exampleTSSr, method = "TPM", tpmLow=0.1)
```

`getTSS` *Precisely identify TSSs from bam files, paired end bam files, bed files, BigWig files, tss files, or tss tables.*

Description

getTSS function is used to precisely identify TSSs from multiple input file formats. The files include users' home-made alignment files (bam format) or downloaded files from public databases. See inputFilesType for details on the supported input file formats.

Usage

```
getTSS(object, sequencingQualityThreshold = 10,
mappingQualityThreshold = 20, softclippingAllowed = FALSE)

## S4 method for signature 'TSSr'
getTSS(
  object,
  sequencingQualityThreshold = 10,
  mappingQualityThreshold = 20,
  softclippingAllowed = FALSE
)
```

Arguments

`object`	A TSSr object.
`sequencingQualityThreshold`	Used only if inputFilesType == "bam" or "bamPairedEnd", otherwise ignored.
`mappingQualityThreshold`	Used only if inputFilesType == "bam" or "bamPairedEnd", otherwise ignored.
`softclippingAllowed`	Used only if inputFilesType == "bam" or "bamPairedEnd". Default is FALSE.

Value

Large List of elements - one element for each sample

Examples

```
data(exampleTSSr)
#getTSS(exampleTSSr)
```

mergeSamples *Merge TSS samples*

Description

Merges individual samples within TSSr object into specified groups.

Usage

```
mergeSamples(object, mergeIndex)

## S4 method for signature 'TSSr'
mergeSamples(object, mergeIndex = NULL)
```

Arguments

object	A TSSr object
mergeIndex	Integer vector specifying which samples to be merged

Value

Large List of elements - one element for each merged sample

Examples

```
data(exampleTSSr)
mergeSamples(exampleTSSr, mergeIndex = c(1,1,2,2))
```

normalizeTSS *Normalize raw TSS counts*

Description

Normalizes raw TSS counts in all samples by tags per million (TPM)

Usage

```
normalizeTSS(object)

## S4 method for signature 'TSSr'
normalizeTSS(object)
```

Arguments

object	A TSSr object.

Value

Large List of elements - one element for each sample

Examples

```
data(exampleTSSr)
normalizeTSS(exampleTSSr)
```

`plotCorrelation` *Pairwise scatter plots and correlations of TSS signal*

Description

Calculates the pairwise correlation coefficients between samples and creates a matix showing pairwise scatter plots and correlation coefficients.

Usage

```
plotCorrelation(object, samples = "all")

## S4 method for signature 'TSSr'
plotCorrelation(object, samples = "all")
```

Arguments

`object`	A TSSr object.
`samples`	Specify samples to be plotted. Can be either "all" to plot all samples in the object or a subset of samples in the object. Default is "all".

Value

pairwise correlations visualized in a graph

Examples

```
data(exampleTSSr)
#plotCorrelation(exampleTSSr, samples = "all")
```

`plotDE` *Plot gene differential expressions*

Description

Vocano plots of gene differential expression (with DESeq2 method) results.

Usage

```
plotDE(object, withGeneName = "TRUE",xlim, ylim)

## S4 method for signature 'TSSr'
plotDE(object, withGeneName = "TRUE", xlim = c(-2.5, 2.5), ylim = c(0, 10))
```

Arguments

object	A TSSr object.
withGeneName	Specify whether to display names for genes which are differentially expressed. Default is "TRUE".
xlim	Only enes of which log2FoldChange value within the xlim range are plotted. Default xlim = c(-2.5, 2.5).
ylim	Only genes of which -log10(pvalue) within the ylim range are plotted. Default ylim = c(0, 10).

Value

gene differential expression visualized in a graph

Examples

```
data(exampleTSSr)
#plotDE(exampleTSSr, withGeneName = "TRUE")
#plotDE(exampleTSSr, withGeneName = "FALSE")
```

---

plotInterQuantile *Plot core promoter interquantile width*

Description

Plots histograms of the interquantile width of processed clusters.

Usage

```
plotInterQuantile(object, samples ="all", tagsThreshold = 1)

## S4 method for signature 'TSSr'
plotInterQuantile(object, samples = "all", tagsThreshold = 1)
```

Arguments

object	A TSSr object.
samples	Specify samples to be plotted. Default is "all".
tagsThreshold	Excludes clusters with tags < tagsThreshold.

Value

core promoter interquantile width visualized in a graph

Examples

```
data(exampleTSSr)
#plotInterQuantile(exampleTSSr, samples = "all")
```

plotShape *Plot core promoter shape*

Description

Plots histograms of core promoter shape scores.

Usage

```
plotShape(object, samples = "all")

## S4 method for signature 'TSSr'
plotShape(object, samples = "all")
```

Arguments

object	A TSSr object.
samples	Specify samples to be plotted. Default is "all".

Value

core promoter shape score visualized in a graph

Examples

```
data(exampleTSSr)
#plotShape(exampleTSSr)
```

plotTSS *Plot TSSs and clusters*

Description

Plots Gviz-track of TSSs, clusters, and genes.

Usage

```
plotTSS(object,samples,tssData = "processed",clusters = "assigned",
clusterThreshold = 0.02,genelist,Bidirection = TRUE,
up.dis =500,down.dis = 500,yFixed = TRUE)

## S4 method for signature 'TSSr'
plotTSS(
  object,
  samples,
  tssData = "processed",
  clusters = "assigned",
  clusterThreshold = 0.02,
  genelist,
  Bidirection = TRUE,
```

```
  up.dis = 500,
  down.dis = 500,
  yFixed = TRUE
)
```

Arguments

object	A TSSr object.
samples	Specify samples to be included for plotting.
tssData	Specify which TSS data to be included for plotting: "raw" or "processed".
clusters	Specify which cluster data to be included for plotting: "all" or "assigned".
clusterThreshold	Ignore downstream clusters if signal < filterClusterThreshold*the strongest clusters within the same gene promoter region. Default value = 0.02.
genelist	List of gene names used for plotting.
Bidirection	Specify whether to display bidirectional TSS signals within defined region. Default is TRUE.
up.dis	Distance upstream of genes to specify plotting range. Default value = 500.
down.dis	Distance downstream of genes to specify plotting range. Default value = 500.
yFixed	Logical, specify whether to fix y axis limits. Default is TRUE.

Value

TSS and cluster examples visualized in a graph

Examples

```
data(exampleTSSr)
#plotTSS(exampleTSSr, samples=c("control","treat"), genelist=c("YBL017C","YBL067C")
#,up.dis =500, down.dis = 500)
```

---

plotTssPCA *Plotting principle component analysis (PCA)*

Description

Calculates principle component analysis (PCA) of all samples and creates a biplot which includes the position of each sample in terms of PC1 and PC2.

Usage

```
plotTssPCA(object, TSS.threshold =10)

## S4 method for signature 'TSSr'
plotTssPCA(object, TSS.threshold = 10)
```

Arguments

object	A TSSr object.
TSS.threshold	Only TSSs with raw signal >= TSS.threshold will be included in PCA analysis

Value

PCA plotted in a graph

Examples

```
data(exampleTSSr)
#plotTssPCA(exampleTSSr)
```

`shapeCluster` *Analysis of core promoter shape*

Description

Calculates core promoter shape based on the distributions of TSSs within core promoters using Shape Index (SI) algorithm (Hoskins et al. 2011) or Promoter Shape Score (PSS) algorithm (Lu et al. 2019).

Usage

```
shapeCluster(object, clusters = "consensusClusters", method = "PSS",
 useMultiCore=FALSE, numCores = NULL)

## S4 method for signature 'TSSr'
shapeCluster(
  object,
  clusters = "consensusClusters",
  method = "PSS",
  useMultiCore = FALSE,
  numCores = NULL
)
```

Arguments

`object`	A TSSr object.
`clusters`	Clusters to be used for calculating shape score: "tagClusters" or "consensusClusters". Default is "consensusClusters".
`method`	Method to be used for calculating core promoter shape score: "SI" or "PSS". Default is "PSS".
`useMultiCore`	Logical indicating whether multiple cores are used (TRUE) or not (FALSE). Default is FALSE.
`numCores`	Number of cores are used in clustering step. Used only if useMultiCore = TRUE. Default is NULL.

Value

Large List of elements - one element for each sample

Examples

```
 data(exampleTSSr)
#shapeCluster(exampleTSSr,clusters = "consensusClusters" , method = "PSS")
#shapeCluster(exampleTSSr,clusters = "tagClusters" , method = "SI")
```

`shiftPromoter` *Select genes which have core promoter shift across different experiments.*

Description

Selects genes which have multiple core promoters and undergo core promoter shifting across different experiments. Generates gene list with Ds (degree of shift) value (Lu et al., 2019), p value and adjusted p value.

Usage

```
shiftPromoter(object, comparePairs, pval=0.01)

## S4 method for signature 'TSSr'
shiftPromoter(object, comparePairs, pval = 0.01)
```

Arguments

`object`	A TSSr object.
`comparePairs`	Specified list of sample pairs for comparison.
`pval`	Genes with adjusted p value >= pval will be returned. Default value = 0.01.

Value

Large List of elements - one element for each sample

Examples

```
data(exampleTSSr)
#shiftPromoter(exampleTSSr,comparePairs=list(c("control","treat")), pval = 0.01)
```

TSSr-class *TSSr: A package for transcription start site sequencing data analyses.*

Description

TSSr is designed to analyze transcription start sites (TSSs) and core promoters with most types of 5'end sequencing data, such as cap analysis of gene expression (CAGE) (Takahashi, Lassmann et al. 2012), no-amplification non-tagging CAGE libraries for Illumina next-generation sequencers (nAnT-iCAGE) (Murata, Nishiyori-Sueki et al. 2014), a Super-Low Input Carrier-CAGE (SLIC-CAGE) (Cvetesic, Leitch et al. 2018), NanoCAGE (Cumbie, Ivanchenko et al. 2015), TSS-seq (Malabat, Feuerbach et al. 2015), transcript isoform sequencing (TIF-seq) (Pelechano, Wei et al. 2013), transcript-leaders sequencing (TL-seq) (Arribere and Gilbert 2013), precision nuclear run-on sequencing (PRO-Cap) (Mahat, Kwak et al. 2016), and GRO-Cap/5'GRO-seq (Kruesi, Core et al. 2013).

Details

TSSr package provides a comprehensive workflow on TSS data starts from identification of accurate TSS locations, clustering TSSs within small genomic regions corresponding to core promoters, and transcriptional activity quantifications, as well as specialized downstream analyses including core promoter shape, cluster annotation, gene differential expression, core promoter shift. TSSr can take multiple formats of files as input, such as Binary Sequence Alignment Map (BAM) files (single-ended or paired-ended), Browser Extension Data (bed) files, BigWig files, ctss files or tss tables. TSSr also generates various types of TSS or core promoter track files which can be visualized in the UCSC Genome Browser or Integrative Genomics Viewer (IGV). TSSr also exports downstream analyses result tables and plots. Multiple cores are supported on Linux or Mac platforms.

Slots

`genomeName` character.

`inputFiles` character.

`inputFilesType` character.

`sampleLabels` character.

`sampleLabelsMerged` character.

`librarySizes` numeric.

`TSSrawMatrix` data.frame.

`mergeIndex` numeric.

`TSSprocessedMatrix` data.frame.

`tagClusters` list.

`consensusClusters` list.

`clusterShape` list.

`refSource` character.

`refTable` data.frame.

`organismName` character.

`assignedClusters` list.

`unassignedClusters` list.

`filteredClusters` list.

`enhancers` list.

`DEtables` list.

`TAGtables` list

`PromoterShift` list.

Index